(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) Publication number: **0 452 989 A2**

(12) EUROPEAN PATENT APPLICATION

(21) Application number: **91110598.9**

(51) Int. Cl.[5]: **G06F 12/08**

(22) Date of filing: **28.05.86**

This application was filed on 26 - 06 - 1991 as a divisional application to the application mentioned under INID code 60.

(30) Priority: **29.05.85 JP 115918/85**
29.08.85 JP 190308/85
27.02.86 JP 42258/86
19.06.85 JP 133432/85

(43) Date of publication of application:
23.10.91 Bulletin 91/43

(60) Publication number of the earlier application in accordance with Art.76 EPC: **0 203 601**

(84) Designated Contracting States:
DE FR GB IT NL

(71) Applicant: **Kabushiki Kaisha Toshiba**
72, Horikawa-cho Saiwai-ku
Kawasaki-shi(JP)

Applicant: **Toshiba Computer Engineering Corporation**
2-9, Suehiro-cho
Oome-shi Tokyo(JP)

(72) Inventor: **Furuya, Akihiko, c/o Patent Division**
Kabushiki Kaisha Toshiba, 1-1 Shibaura 1-chome
Minato-ku, Tokyo 105(JP)
Inventor: **Kanamaru, Kouiti, c/o Patent Division**
Kabushiki Kaisha Toshiba, 1-1 Shibaura 1-chome
Minato-ku, Tokyo 105(JP)
Inventor: **Inoue, Junichi**
3-12, Minami-cho
Hachioji-shi, Tokyo(JP)

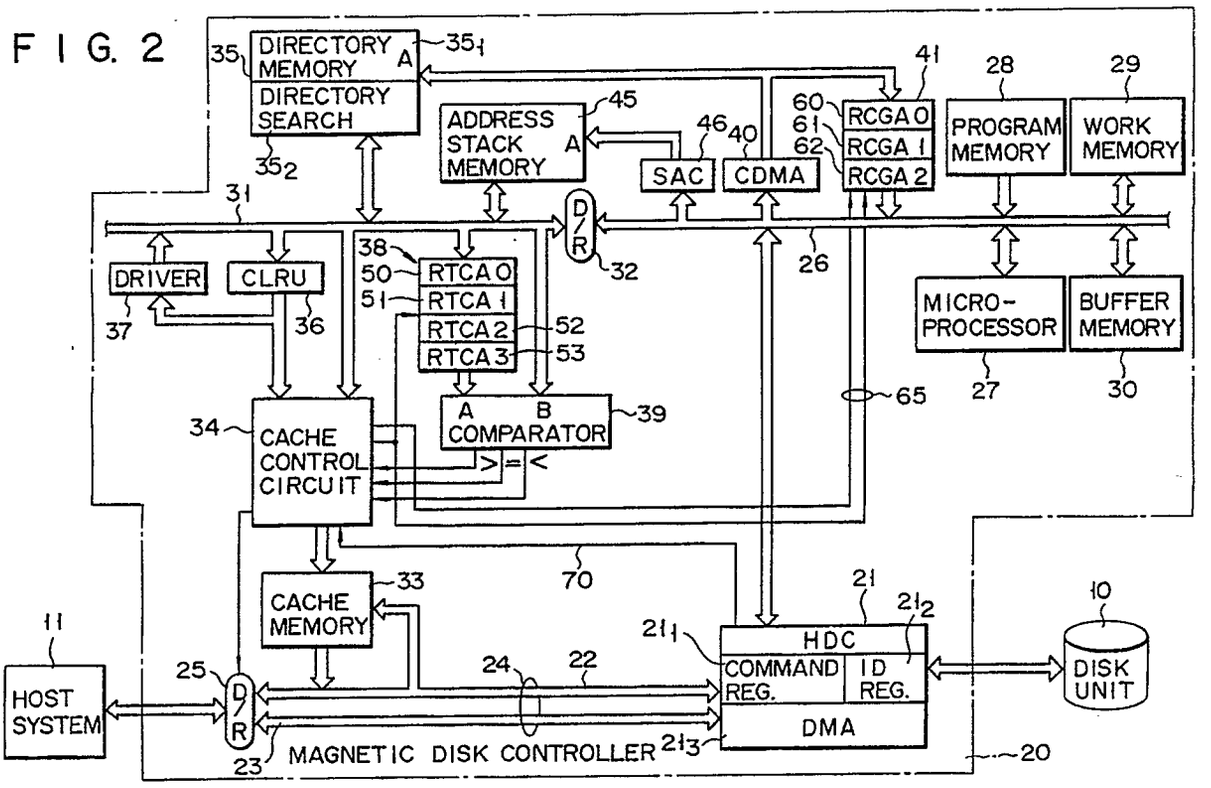
(74) Representative: **Henkel, Feiler, Hänzel & Partner**
Möhlstrasse 37
W-8000 München 80(DE)

(54) Magnetic disk controller incorporating a cache system adopting an LRU system.

(57) The present invention provides a magnetic disk controller incorporating a cache system having a cache memory. In order to improve the operation of the cache memory the magnetic disk controller comprises cache operation designation data producing means (45) or uses a data transfer direction control code in cache operation designation data representing five modes of operations.

EP 0 452 989 A2

FIG. 2


DIRECTORY MEMORY
DIRECTORY SEARCH
ADDRESS STACK MEMORY
SAC
CDMA
RCGA 0
RCGA 1
RCGA 2
PROGRAM MEMORY
WORK MEMORY
D/R
DRIVER
CLRU
RTCA 0
RTCA 1
RTCA 2
RTCA 3
MICRO-PROCESSOR
BUFFER MEMORY
A B COMPARATOR
CACHE CONTROL CIRCUIT
CACHE MEMORY
HDC
COMMAND REG.
ID REG.
DMA
HOST SYSTEM
DISK UNIT
MAGNETIC DISK CONTROLLER

The present invention relates to a magnetic disk controller incorporating a cache system adopting an LRU (Least Recently Used) system in a replacement algorithm for cache blocks.

Disk cache systems capable of high-speed data access to magnetic disks have been developed. A disk cache is a memory device using a semiconductor RAM and is designed to eliminate an access gap between a high-speed main memory and low-speed large-capacity secondary memories (i.e., magnetic disk units). The disk cache is normally arranged between the main memory and a magnetic disk unit, in a magnetic disk controller, and serves as a data buffer.

The principle of the disk cache is the same as that of a central processing unit (CPU) cache. If a CPU first accesses part of a file, necessary data blocks are stored on a disk. In this case, the disk cache does not contribute to such CPU access.

Necessary blocks are transferred from the disk to the main memory. At the same time, they are written in the disk cache. If the CPU subsequently accesses the same blocks, they are transferred from the disk cache, but not from the disk. The access time of the magnetic disk unit is normally 20 to 30 ms, while the access time of the disk cache is about 1 to 3 ms. The access time can, therefore, be decreased to about 1/10. When a total access time is regarded as an input/output response time including data transfer time and channel wait time, it can be normally decreased to 1/2 to 1/3.

Since a disk cache capacity is smaller (about 1/1000) than that of the disk, all file blocks required by the CPU are not always stored in the disk cache. In order to load new blocks when the disk cache is filled with file blocks, old blocks must be removed from it. When accessing of the cache allows retrieval of necessary data from the cache, it is called a "hit", and when retrieval of necessary data cannot be performed, it is called a "miss". The average hit times per times of the difference of the cache is called a hit ratio.

In addition to the disk cache, disk cache mechanism components are a disk cache handler as a program in the input/output controller, a data transfer mechanism between the main memory and the disk cache, and a disk cache directory. The disk cache handler cooperates with an operating system (OS) to determine the hit/miss ratio of the disk cache, control of data transfer between the main memory, the disk cache, and the magnetic disks, and acquisition of data representing utilization efficiency of the disk cache and the hit ratio, and so on.

Write access schemes include a write through scheme and a write back scheme. The write through scheme is called a store through scheme for writing data directory in a disk. The cache hit processing is performed in such a way that the cache hit block is disabled by not writing data therein, or the data is written in the cache hit block after it is directly written into the disk.

The write back scheme is called a store in, copy back, or post store scheme. According to this scheme, the disk cache write is the highest priority. When data write into a file in the disk cache is completed, data input/output is completed. If the file is unnecessary or the cache block storing the file must be emptied for other files, data is transferred from the cache to the disk.

In the disk cache, data copying from the magnetic disk to the cache memory is called staging, and copying from the cache memory to the magnetic disk is called destaging.

A typical disk cache memory mapping scheme is a set associative scheme. According to this scheme, the magnetic disk and the disk cache memory are divided into a plurality of blocks (e.g., one block = one track = 64 sectors). Each block belongs to any one of columns. In order to transfer a given block from the magnetic disk to the cache memory, this block is written in an identical column of the cache memory and data (representing which block, i.e., a block number, in the magnetic disk is stored in the cache memory) is stored in the directory in units of columns. By referring this directory, it is possible to determine whether specific data is stored in the cache memory or not. If the data is stored in the cache memory, it can be read out, not from the magnetic disk unit but from the cache memory.

The disk cache replacement algorithm is an LRU algorithm. LRU stands for "Least Recently Used." According to the LRU algorithm, the block which stays in the cache for the longest period after the last access is selected as the least necessary block. In a disk cache having no empty space, if the cache memory must be assigned for new staging, the least necessary track data is removed to obtain an empty space. The LRU algorithm is used to assure this empty space.

If a cache hit occurs as a result of the directory search, the LRU value in the cache directory corresponding to the hit cache block is initialized to "0". In the case of cache miss, the LRU value corresponding to the cache block to be replaced must be registered again. In this case, if the cache miss occurs in the read request mode, the corresponding data block is read out from the magnetic disk unit and is stored in the cache block in the cache memory which is to be replaced.

In a conventional cache system, LRU setting values are determined regardless of the data blocks stored in the cache memory.

The frequency of use of the data blocks stored

in the cache blocks varies according to different types of files. Once a data block having a low frequency of use is stored in the cache block, the resident possibility of that data block in the cache memory is often higher than that of a file having a high frequency of use. Because of this, a decrease in hit rate of the cache memory tends to occur.

In the conventional disk cache system of this type, disk write operation is always performed in response to a disk write instruction from a host system. According to the conventional system, even if the data in the disk area subjected to disk write is stored in the cache memory (even if cache hit occurs), the contents of the cache memory and the disk unit are simultaneously updated. This is not to allow non-updated data to be left in the disk when a failure, such as a power failure, occurs.

According to the system described above, however, high performance of the disk cache structure can be achieved only when a read frequency is high. If the write frequency is high, a great deal of improvement in performance cannot be expected.

When a power unit free from the effects of power failure is used, disk write need not be performed at the time of cache hit in the write request mode. However, a power unit of this type is very expensive.

Assume that a read access request for a given file is sent from the CPU. The magnetic disk controller retrieves a disk cache directory (a table having file information such as a file name, a file size, and a file address) and checks if the data block of interest is stored in the disk cache. I fache hit occurs, the corresponding data block is transferred from the cache memory to the main memory.

In the case of cache miss, however, several blocks (usually those for one track) including the required block are sent to the cache memory, and only the required block is sent to the main memory. Data transfer to the cache memory is performed simultaneously with that to the main memory. However, the blocks can be transferred to the cache memory and the block of interest can be transferred to the main memory thereafter.

A large demand has arisen for a magnetic disk controller for satisfying the above-mentioned two transfer schemes and for performing effective data transfer control if cache miss occurs.

If a cache memory is commonly used by, e.g., 4 magnetic disk drives DK0 through DK3, different cache memory areas are permanently assigned for respective drives DK0 through DK3, and a directory memory having a corresponding memory allocation is prepared, as shown in Fig. 1.

According to the system described above, however, the limited area of the cache memory is divided, and the cache locations respectively assigned to the magnetic disk drives are decreased, leading to a reduction in the cache hit rate. If the access frequency varies for the different drives, the frequency of use of the cache memory location corresponding to the drive having the lowest access frequency is decreased. As a result, the overall utilization efficiency of the cache memory is degraded.

It is object of the present invention to provide a magnetic disk controller for effectively performing data transfer control upon occurrence of read errors and for achieving high-speed transfer.

It is still another object of the present invention to provide a magnetic disk controller capable of effectively using a cache memory in common for a plurality of magnetic disk drives, regardless of their number.

In order to achieve the above objects of the present invention, there is provided a magnetic disk controller as stated in claim 1 or 2.

The magnetic disk controller of the present invention uses a cache system wherein setting values for LRU initialization or re-registration are variables and the resident possibility of data blocks in the cache memory can be controlled, and wherein the frequency of date written in the disk can be decreased without using an expensive power source unit to improve system performance.

This invention can be more fully understood from the following detailed description when taken in conjunction with the accompanying drawings, in which:

Other objects and features of the present invention will be apparent from the following description taken in connection with the accompanying drawings, in which:

Fig. 1 is a schematic view showing the correspondence between a cache memory and a directory memory in a conventional system;

Fig. 2 is a block diagram showing an embodiment of a magnetic disk controller incorporating a disk cache memory according to the present invention;

Fig. 3 is a format showing an entry structure of a directory memory in Fig. 2;

Fig. 4 is a format of a read/write command used in the system of Fig. 2;

Fig. 5 is a format of an LRU setting value;

Fig. 6 is a format of directory data stored in directory memory 35_1 according to another embodiment of a disk cache system;

Fig. 7 is a format showing the main part of a disk write instruction employed in the format of Fig. 6;

Fig. 8 is a format showing the main part of a disk read instruction used in the format of Fig. 6;

Figs. 9 through 12 are flow charts for explaining

the operation of the format in Fig. 6, in which Fig. 9 is a flow chart for explaining control of magnetic disk controller 20 when the disk write instruction is supplied thereto from a host system, Fig. 10 is a flow chart of the search subroutine, Fig. 11 is a flow chart for explaining control of magnetic disk controller 20 when the disk read instruction is supplied thereto from the host system, and Fig. 12 is a flow chart for explaining the operation of magnetic disk controller 20 during a power failure;

Figs. 13A through 13E are views for explaining the operation contents of data transfer direction control code C used in the magnetic disk controller according to the present invention;

Fig. 14 is a chart for explaining cache operation designation data produced upon occurrence of a read error;

Fig. 15 is a timing chart for explaining the disk read operation in Fig. 14;

Fig. 16 is a chart for explaining another cache operation designation data;

Fig. 17 is a block diagram showing another embodiment of a magnetic disk controller according to the present invention;

Fig. 18 is a format of directory data used in the embodiment of Fig. 17;

Fig. 19 is a format of a read/write command used in the embodiment of Fig. 17;

Fig. 20 is a data format of a directory memory used in the embodiment of Fig. 17; and

Fig. 21 is a schematic view showing the correspondence between cache memory 33 and directory memory 35_1 in the embodiment of Fig. 17.

Fig. 2 is a block diagram showing an embodiment of magnetic disk controller 20 incorporating a disk cache system. Hard disk control circuit (to be referred to as an HDC hereinafter) 21 for controlling disk unit 10 is connected to host system 11 through first bus 24, which consists of data line 22 and control line 23, and driver/receiver (to be referred to as a D/R hereinafter) 25. HDC 21 is also connected to second bus 26. HDC 21 consists of command register 21_1, ID register 21_2, and DMA 21_3. Bus 26 is also connected to microprocessor 27 for controlling the overall operation of controller 20, program memory 28 storing control programs (microprograms) for microprocessor 27, work memory 29 used as a work area of microprocessor 27, and buffer memory 30 for compensating a difference between the transfer rates of disk unit 10 and host system 11. Third bus 31 is connected to cache control circuit 34, mainly for controlling data transfer, the operation of which includes control of cache memory 33, directory memory/search circuit 35 consisting of directory memory 35_1 and directory search circuit 35_2, and address stack memory 45. Memory 45 is a FIFO (first-in first-out) stack for storing cache operation designation data consisting of three parameters. These three parameters are transfer start cache address $\underline{a}$, the number of transfer sectors ℓ, and data transfer direction control code C. The output of stack address counter 46 is connected to address port A of address stack memory 45. Bus 26 is connected to the input of counter 46.

Bus 31 is connected to the input to LRU counter (referred to as an CLRU hereinafter) 36 for storing an LRU value read out from memory 35_1 in the directory search mode, as well as the output of driver 37. The output of CLRU 36 and the input of driver 37 are connected to cache control circuit 34. Bus 31 is also connected to the input port of register file 38 and the B input of comparator 39. The A input of comparator 39 receives the data from bus 31. The output port of comparator 39 is connected to control circuit 34. Control circuit 34 is connected to register file 38 to control inputs thereto, i.e., the CLRU 36 contents (LRU values) output by driver 37 on bus 31. File 38 comprises four registers 50 through 53, i.e., RTCA0 through RTCA3.

The output of directory memory address counter (to be referred to as a CDMA hereinafter) 40 is connected to address port A of directory memory 35_1 to designate the addresses of memory 35_1. The input of CDMA 40 is connected to second bus 26. The output of CDMA 40 is connected to the input of register file 41. The output of file 41 is also connected to bus 26. File 41 is connected through control line 65 so as to receive control signals from cache control circuit 34. Register file 41 consists of three registers 60 through 62, i.e., RCGA0 through RCGA2.

Each entry of directory memory 35_1 includes disk block number field (8 bits) 81 representing the number of data blocks registered in cache memory 33, and LRU counter field (8 bits) 83 representing an LRU value, as shown in Fig. 3. The number of entries in memory 35_1 is equal to that of cache blocks in cache memory 33. In this embodiment, 16 entries are given for each column. One track of disk unit 10 consists of 64 sectors, each corresponding to one block of cache memory 33. The area of disk unit 10 is divided into column 0 and column 1 which store data in units of 64 sectors. A column value is represented by the seventh bit from the LSB of a disk logical address (to be described later). The disk block number is represented by the MSB excluding the lower seven bits of the disk logical address.

The magnetic disk controller of this embodiment has the following features:

(i) One sector of the magnetic disk consists of 256 bytes.

(ii) One block (cache block) of cache memory 33 consists of 16K bytes (64 sectors) and therefore one block (disk block) of the disk consists of 16K bytes (64 sectors).

(iii) A mapping scheme of cache memory 33 is a set associative scheme.

(iv) A cache block replacement scheme is an LRU scheme.

(v) The number of levels (i.e., the number of entries) of directory memory 35_1 is 16 per column.

(vi) The capacity of cache memory 33 is 512 Kbytes.

The operation of the embodiment shown in Fig. 2 will be described below.

A read/write command from host system 11 is stored in command register 21_1 in HDC 21 through D/R 25 and bus 24. The read/write command is a 6-byte instruction including command code 85 for designating read/write operation or the like, a disk logical address consisting of logical address 2 (MSB) 87, logical address 1 89, and logical address 0 (LSB) 91, the number of data blocks, and each field 93 of cache control bits CC2 through CC0 (Fig. 4).

The read/write command in register 21_1 is read out by microprocessor 27 and stored in a predetermined area of work memory 29. When microprocessor 27 stores the read/write command from host system 11 in memory 29, microprocessor 27 issues a seek command which is then set in command register 21_1 in HDC 21. HDC 21 controls disk unit 10 to seek a specified track. Microprocessor 27 also sets, in accordance with the read/write command, a disk block number corresponding to the access area of disk unit 10, and an LRU value of "0" in register file 38 through bus 26, D/R 32, and bus 31. Microprocessor 27 sets bit 6 of logical address 0 (Fig. 4), i.e., the start address of the currently read-accessed entry in each area (for 16 entries) in directory memory 35_1, in CDMA 40. In this case, each area is accessed by the column value. The following directory search cycle is repeated 16 times.

In the directory search cycle, the address specified by CDMA 40 is set in RCGA0 60 in register file 41. Therefore, the start position of the entry currently searched is always stored in RCGA0 60 in register file 41.

CDMA 40 is sequentially counted up. Directory data of the entry in directory memory 35_1 is read out onto bus 31 in units of bytes in response to the count from the CDMA 40. The readout directory data is supplied to the B input of comparator 39. Byte data from register file 38 is supplied to the A input of comparator 39 in response to the data supplied to the B input. Comparator 39 compares the contents of the A and B inputs.

The comparison result of comparator 39, i.e., the result of comparison of the target data block number from file 38 with the registered data block number from memory 35_1 in units of bytes, is supplied to cache control circuit 34. The result of comparison between the LRU value from file 38 and that from memory 35_1 is also supplied to cache control circuit 34. The LRU value supplied from memory 35_1 to the B input of comparator 39 through bus 31 is supplied to CLRU 36. The LRU value stored in CLRU 36 is incremented by one.

Cache control circuit 34 determines if cache hit occurs according to the data block number comparison result from comparator 39. If circuit 34 determines that coincidence is detected by comparator 39 and cache hit occurs, it appends a cache hit flag to the address set in RCGA0 60 in register file 41, i.e., to the address representing the start position of the entry being searched in the current directory search cycle. The address with a cache hit flag is set in RCGA1 61 in register file 41. Cache control circuit 34 determines the entry of directory memory 35_1 which corresponds to the cache block to be replaced at this time, according to the LRU value comparison result. More specifically, if the LRU value from memory 35_1 exceeds that stored in file 38, cache control circuit 34 determines that the cache block corresponding to the entry is a candidacy of block to be replaced. Control circuit 34 transfers the address from RCGA0 60 in file 41 to RCGA2 62 therein. Control circuit 34 causes driver 37 to send the incremented LRU value from CLRU 36 onto bus 31. The LRU value in register file 38 is then updated to correspond with that appearing on bus 31. Cache control circuit 34 also writes the LRU value stored in CLRU 36 via bus 31 in the corresponding entry in directory memory 35_1, regardless of the comparison result of comparator 39. In this embodiment, the LRU value in the corresponding entry in memory 35_1 is incremented by one for each directory search cycle.

The above operation is the directory search cycle. This cycle is repeated 16 times to complete the directory search corresponding to the given read/write command. In this case, if cache hit is detected, the start address of the entry in memory 35_1 which corresponds to the hit cache block is set in RCGA1 61 in register file 41. The start address of the entry in memory 35_1 which has the largest LRU value in the specified column is stored in RCGA2 62.

When the directory search is completed, microprocessor 27 fetches the content of RCGA1 61 in file 41 through bus 26, and checks the flag mentioned above in RCGA1 61 whether cache hit or miss occurs. If microprocessor 27 detects cache hit, the LRU value in the entry in memory 35_1

which is specified by the content (directory address) of RCGA1 61 in file 41 is initialized (or re-registered). However, if the read mode is specified, even if cache miss is detected, the LRU value in the entry in memory 35_1 which is specified by the content of RCGA2 62 in file 41 is initialized (or re-registered).

In initialization (or re-registration) of the LRU value, microprocessor 27 retrieves the read/write command sent from host system 11 and retained in buffer memory 30. Microprocessor 27 checks if cache control bit CC2 among bits CC2 through CC0 is "0". If CC2 = "0", then CC1 and CC2 are defined as LRU setting data. In this case, CC1 and CC2 are shifted to the left by 6 bits to prepare an LRU setting value "CC1 (MSB), CC0, 0, 0, 0, 0, 0, 0 (LSB)", as shown in Fig. 5. The same LRU setting value is supplied to directory memory 35_1 through bus 26, D/R 32, and bus 31, and is registered in LRU counter field 83 of the entry in memory 35_1. In LRU value comparison and setting performed by CLRU 36 in the next directory search, the updated LRU setting value is used. If CC1 and CC0 are set to be "1", the resident period of the data block in cache memory 33 can be decreased. However, if CC1 and CC0 are set to be "0", the period can be increased.

Figs. 6 through 12 show another disk cache system wherein the number of write cycles for the disk can be decreased without using an expensive power source, thereby improving system performance.

Fig. 6 shows a format of directory data stored in directory memory 35_1.

Directory data set in each entry in directory memory 35_1 consists of a cache active bit representing whether the content of a block (cache block) in cache memory 33 for a corresponding entry is effective or not, a write command receiving bit representing whether the write data specified by host system 11 is completely written in disk unit 10 or not, a resident bit representing whether replacement of the cache block corresponding to this entry is inhibited, a disk address (disk block address) of a data block (disk block) stored in cache memory 33, and a replacement counter (replacement counter value) used for replacement operation.

Fig. 7 shows a format of a disk write instruction sent from host system 11 to disk controller 20. The write instruction includes a specified write instruction designation bit for determining whether the write instruction is the specified write instruction which allows data write in cache memory 33 but does not require data write in disk unit 10 when cache hit is detected in the disk write request mode.

Fig. 8 shows a format of a disk read instruction sent from host system 11 to disk controller 20. The read instruction includes a resident designation bit for designating setting/resetting of a resident bit of an entry (called the replacement entry) in memory 35, which corresponds to the cache block (subjected to replacement) in cache memory 33 when cache miss is detected in the disk read request mode.

The operation of the system described above will be described below. The operation is exemplified by a case with reference to the flow chart of Fig. 9 wherein a disk write instruction is sent from host system 11 to disk controller 20.

When the disk write instruction is sent from host system 11 to disk controller 20, controller 20 searches directory memory 35_1 (step 101). In the search step, as shown in Fig. 10, microprocessor 27 checks whether data to be written in the block of disk unit specified by a command exist in cache memory 33 (step 103). In the case of a write command (step 104), the number of entries whose write command receiving bits are set is counted (step 105). The count from the replacement counter is stored in RTCA0 register 50 in register file 38 (step 106).

In step 107 of Fig. 9, if microprocessor 27 detects cache miss, controller 20 writes the write data from host system 11 directly in disk unit 10 (step 109), and the flow is ended. However, if YES in step 107, disk controller 20 checks if the specified write instruction designation bit in the disk write instruction from host system 11 is set, i.e., if the disk write instruction is the specified write instruction (step 111).

If NO in step 111, controller 20 writes the write data from host system 11 in cache memory 33 and the disk unit 10 (step 113). Controller 20 then updates the replacement counter in the bit entry in directory memory 35_1 (step 115), and the flow is ended.

However, if YES in step 111, controller 20 retrieves the contents of RTCA0 50 to check if there are N or more entries whose write command receiving bits are set, i.e., if there are N or more blocks which have been written in cache memory 33 but not yet in disk unit 10 (step 117).

If NO in step 117, controller 20 writes the write data from host system 11 in cache memory 33, but does not write it in unit 10 (step 119). Controller 20 sets the write command receiving bit in the hit entry in directory memory 35_1 (step 121), and the flow is ended.

However, if the number of blocks whose data are not written in unit 10 is N or more, controller 20 3rites the write data from host system 11 in cache memory 33 and disk unit 10 (step 123). Controller 20 writes certain of the blocks not written in disk unit 10 in memory 33, i.e., those which correspond

to the directory memory 35_1 entries whose write command receiving bits are set (step 125). In this case, the number of blocks to be written in unit 10 is determined to be smaller than N (e.g., N - 2). The operation in step 125 is performed such that the number of blocks not written in disk unit 10 is smaller than the maximum number N of blocks to be written in disk unit 10 in a back-up operation mode, i.e., at the time of power failure. Controller 20 then resets write command receiving bits of directory memory 35_1 entries corresponding to the cache blocks written in unit 10 (step 127), and the flow is ended.

If the cache hit is detected in the case of the disk write instruction whose specified write instruction designation bit is set, the corresponding write data is registered in cache memory 33 but not in disk unit 10, thereby decreasing the write cycles of the disk unit 10. The number of blocks written in cache memory 33 but not in unit 10 is controlled to be smaller than N. At the time of power failure, the blocks not written in unit 10 are then written therein (to be described later).

An operation will be described with reference to the flow chart in Fig. 11, wherein a disk read instruction is sent from host system 11 to disk controller 20.

When the disk read instruction is sent from system 11 to controller 20, controller 20 searches directory memory 35_1 (step 129) to check if the data in unit 10 specified by the instruction exists in cache memory 33, i.e., if cache hit is detected (step 131).

If YES in step 131, controller 20 reads out the target data from the cache block in cache memory 33 so as to correspond to the hit entry in directory memory 35_1. The readout data is sent to host system 11 (step 133). Controller 20 sets or resets the resident bit of hit entry in memory 35_1 according to the content of the resident designation bit of the disk read instruction (step 135), and the flow is ended. The replacement counter area in each hit entry is updated in step 135.

However, if NO in step 131, controller 20 checks if all resident bits of the entries in directory memory 35_1 are set (step 137).

If YES in step 137, controller 20 determines that no cache blocks to be replaced exist and transfers the read data from unit 10 to only host system 11 (step 139). In this case, the read data is not written in cache memory 33.

However, if there is even one directory memory entry whose resident bit is not set, controller 20 determines the entry to be replaced according to the LRU algorithm (i.e., the content of the replacement counter) (step 141). The cache block corresponding to this entry is detected as the cache block to be replaced. Subsequently, disk controller 20 checks if the write command receiving bit of the entry to be replaced is set (step 143).

If YES in step 143, controller 20 writes the cache block corresponding to this entry, i.e., the data of cache block to be replaced, in disk unit 10 (step 145). Controller 20 then transfers read data from unit 10 (specified by the disk read instruction from host system 11) to system 11 and performs data write in the cache block data to be replaced (step 147). Controller 20 sets a disk block address and the initial value for the replacement counter in the entry (the entry to be replaced) in directory memory 35_1 which specifies the cache block. At the same time, controller 20 resets the write command receiving bit and sets or resets the resident bit according to the content of the resident designation bit of the disk read instruction from host system 11 (step 149).

The cache block corresponding to the directory memory 35_1 entry whose resident designation bit is set in step 149 is excluded from the blocks subjected to replacement. This block data, i.e., the data of the disk block specified by the disk read instruction resides in cache memory 33 at least until the resident bit is reset (i.e., until resident cancel of the cache block data) in response to the resident designation bit in the disk read instruction. The disk block residence can be specified in response to the resident designation bit of the disk read instruction, or can be performed with software. This operation is used for high-speed jobs, thereby improving overall system performance.

An operation at the time of power failure will be described with reference to the flow chart in Fig. 12.

Power failure is detected by a cutoff detector (not shown). A power source is designed to be operated for a predetermined period of time after the cutoff detector detects a power failure. During this back-up period, disk controller 20 performs the following operation. If the cutoff detector detects a power failure, controller 20 interrupts the currently executed processing (step 151). Controller 20 accesses directory memory 35_1 and sequentially writes cache block data (cache block data not written in disk unit 10) corresponding to the entry whose write command receiving bit is set (step 153).

The number of cache blocks corresponding to the entries whose write command receiving bits are set is smaller than the maximum number N of blocks to be written in unit 10 during the back-up period after the power failure. Therefore, all block data not written in unit 10 can be properly written therein, in step 153. At the time of system initialization after the power failure, data can be read from unit 10 without read errors. It is possible to provide most of the non-written blocks as continuous resi-

dent blocks in the disk area upon resident designation of the disk write instruction. This indicates that high-speed write of non-written blocks in unit 10 can be performed and the N value can be increased.

Still another embodiment for effectively performing data transfer control in the case of read errors will be described with reference to Fig. 2 and Figs. 13A through 15.

The disk access request from host system 11 is held in command register 21_1 in HDC 21 through D/R 25 and bus 24. This request is read out by microprocessor 27 and stored in a predetermined area of work memory 29. At the same time, microprocessor 27 supplies a seek command to HDC 21. HDC 21 controls disk unit 10 to seek a track specified by the seek command. Microprocessor 27 supplies the disk block number corresponding to the disk unit area subjected to data access and the directory memory address to directory search circuit 35_2 through bus 26, D/R 32, and bus 31. Microprocessor 27 initializes circuit 35_2. Search circuit 35_2 performs directory search using the disk block number as the target. The directory search result is held in directory memory/search circuit 35.

Microprocessor 27 produces at least one cache operation designation data according to the disk block number held in directory memory/search circuit 35 and the contents of the disk access request. The produced data is stored in address stack memory 45 through bus 26, D/R 32 and bus 31.

Data transfer direction control code C in the cache operation designation data represents five modes of operation. These five modes will be described with reference to Figs. 13A through 13E.

(i) C = 1

This is used for cache hit in the disk read request mode. In this case, the contents of cache memory 33 are transferred to host system 11, and disk unit 10 is not accessed, as shown in Fig. 13A.

(ii) C = 2

This is used for cache miss in the disk read request mode. As shown in Fig. 13B, disk unit 10 is read-accessed, and the readout data is transferred to host system 11 through buffer memory 30 and, at the same time, is written in cache memory 33.

(iii) C = 3

This is used for cache miss in the disk read request mode and a sector is not requested by the host system but exists in the cache block (undesired sectors may often be written since a write operation is performed in units of blocks). As shown in Fig. 13C, the read data is written in cache memory 33 but is not transferred to host system 11.

(iv) C = 4

This is used for cache hit in the disk write request mode. As shown in Fig. 13D, the write data from host system 11 is transferred to disk unit 10 through buffer memory 30 and, at the same time, is written in cache memory 33.

(v) C = 5

This is used for cache miss in the disk write request mode. The data from host system 11 is written in only disk unit 10 through buffer memory 30, but not in cache memory 33.

The production of cache operation designation data will be described below. Assume that a disk access request from host system 11 represents disk read for 6 sectors starting from sector number 10, and that cache miss occurs during the disk read. Also assume that one track of the disk consists of 64 sectors (sector 0 through sector 63), and that one block of cache memory 33 consists of 64 sectors.

When microprocessor 27 detects the cache miss in read operation according to the directory search result of directory memory/search circuit 35, microprocessor 27 read-accesses the ID field of the disk. The readout data is latched in ID register 21_2 in HDC 21 to read the sector number from the ID field.

Microprocessor 27 subdivides the updating cache block into subunits by use of cache addresses each corresponding to the sector next to that of the read sector number, the start sector of the disk area subjected to transfer to the host system, and the end sector. Cache operation designation data are prepared for the respective subunits. Assuming that the read sector number is 4, the cache block is subdivided into subunit A corresponding to the disk area of sector numbers 0 through 4, subunit B of sector numbers 5 through 9, subunit C of sector numbers 10 through 15, and subunit D of sector numbers 16 through 63. When data is read from sector number 5, i.e., when data transfer is performed in an order of B, C, D, and A, effective data transfer without awaiting disk rotation can be achieved.

The cache block shown in Fig. 14 shows one block in cache memory 33. The cache block is subdivided into subunits A, B, C, and D. Of course, the cache block is not physically divided into subunits. Numeric values 0 to 63 represent the disk sector numbers. Data of disk sector numbers 5 to

9, for example, is written in subunit B starting with address aB.

Microprocessor 27 first produces cache operation designation data for subunit B and pushes down the data in address stack memory 45 through bus 26, D/R 32, and bus 31. As shown in Fig. 14, the cache operation designation data for subunit B consists of transfer start cache address aB for subunit B, the number ℓB of transfer sectors, and the data transfer direction control code (C = 3).

Microprocessor 27 then produces cache operation designation data for subunit C and pushes it down in address stack memory 45. The cache operation designation data for subunit C consists of transfer start cache address aC, the number ℓC of transfer sectors, and the data transfer direction control code (C = 2), as shown in Fig. 14.

Subsequently, microprocessor 27 produces cache operation designation data for subunit D and pushes it down in memory 45. The cache operation designation data for subunit D consists of transfer start cache address aD, the number ℓD of transfer sectors, and the data transfer direction control code (C = 3), as shown in Fig. 14.

Finally, microprocessor 27 produces cache operation designation data for subunit A and pushes it down in memory 45. The cache operation designation data for subunit A consists of transfer start cache address aA, the number ℓA of transfer sectors, and the data transfer direction control code (C = 3), as shown in Fig. 14.

When microprocessor 27 completes production of cache operation designation data and its storage in address stack memory 45, microprocessor 27 sends a control signal to HDC 21 to designate disk read of 59 sectors starting from sector number 5, i.e., the disk read from the start sector of the disk area corresponding to subunit B to the last sector of the corresponding track. Microprocessor 27 sets the start address for address stack memory 45 in stack address counter 46, and starts cache control circuit 34.

HDC 21 receives the disk read command from microprocessor 27 and data of the specified area from disk unit 10 and stores them in buffer memory 30 through bus 26. When disk read data of one or more sectors is buffered in memory 30, DMA 21_3 in HDC 21 sends a data transfer request to host system 11 through control line 23, and, at the same time, a data transfer request to cache control circuit 34. When DMA 21_3 receives the response signals from host system 11 and cache control circuit 34, it reads the data from buffer memory 30 and sends it onto data line 22. This read operation is time-divisionally performed with storage in memory 30.

When cache control circuit 34 is initialized by microprocessor 27, circuit 34 reads the specified data from the area (corresponding to subunit B of Fig. 14) of memory 45 which is accessed by stack address counter 46. Cache control circuit 34 sends address and write signals to cache memory 33, and sequentially writes data output from DMA 21_3 on line 22 in the specified area of memory 33 in synchronism with data transfer by DMA 21_3. In this case, cache control circuit 34 controls D/R 25 to inhibit data transfer to host system 11 via line 22.

When cache control circuit 34 completes data write for subunit B, it controls stack address counter (SAC) 46. In other words, when address aB is updated by address ℓB, SAC 46 is incremented by one. The next cache operation designation data, i.e., that for subunit C, is read from address stack memory 45 and designated control is performed. More specifically, the data sequentially sent from DMA 21_3 onto line 22 are sequentially written in subunit C of the cache memory 33. At the same time, D/R 25 is controlled to transfer data from line 22 to system 11. When the data write of subunit C is completed, cache control circuit 34 performs control specified by the next cache operation designation data, i.e., the cache operation designation data for subunit D. Control circuit 34 writes data from line 22 in the designated subunit D of cache memory 33 and, at the same time, controls D/R 25 to inhibit write operation of data from line 22 to system 11.

After HDC 21 completes the write operation for subunits B, C, and D, it detects an index pulse, as shown in Fig. 15, and then interrupts microprocessor 27. Upon detection of the interrupt from HDC 21, microprocessor 27 desigates disk read for subunit A (sector 0 through sector 4). HDC 21 receives data of disk area corresponding to subunit A, and stores it in buffer memory 30. The data buffered in memory 30 is sent by DMA 21_3 in HDC 21 onto line 22, and is written in subunit A of cache memory 33 under the control of cache control circuit 34. In this case, data transfer to system 11 via line 22 is inhibited.

In the embodiment shown in Fig. 14, the disk access request from host system 11 represents cache miss in the disk read of 6 sectors, starting from sector number 10, and the current read sector number is 4.

In still another embodiment of the present invention, shown in Fig. 16, a disk access request from host system 11 represents disk read of 6 sectors, starting from sector number 10, and the current read sector number is 12. In this case, the cache block is subdivided into subunit A corresponding to a disk area of sector numbers 0 through 9, subunit B corresponding to sector numbers 10 through 12, subunit C corresponding to sector numbers 13 through 15, and subunit D corresponding to sector numbers 16 through 63. In

this case, if data is read from sector number 13, i.e., if data transfer is performed in an order of C, D, A, and B, effective data transfer without awaiting disk rotation can be achieved.

Microprocessor 27 first produces cache operation designation data for subunit C and pushes it down in address stack memory 45. Cache operation designation data for subunit C consists of transfer start cache address aC, the number ℓC of transfer sectors, and the data transfer direction control code (C = 2).

Microprocessor 27 produces cache designation data for subunit D and pushes it down in address stack memory 45. This cache operation designation data consists of transfer start cache address aD, the number ℓD of transfer sectors, and the data transfer direction control code (C = 3).

Microprocessor 27 then produces cache operation designation data for subunit A and pushes it down in address stack memory 45. This cache operation designation data consists of transfer start cache address aA, the number ℓA of transfer sectors, and the data transfer direction control code (C = 3).

Finally, microprocessor 27 produces cache operation designation data for subunit B and pushes it down in address stack memory 45. This cache operation designation data consists of transfer start cache address aB, the number ℓB of transfer sectors, and the data transfer direction control code (C = 2), as shown in Fig. 16.

Data can be read in an order of subunits C, D, A, and B under the control of cache control circuit 34, as in the embodiment of Fig. 14.

According to this embodiment, since data transfer control is performed according to the cache operation designation data consisting of the transfer start cache address, the number of transfer sectors, and the data transfer direction control code, data transfer can be effectively performed upon occurrence of a read error caused by storage of data not requested by the host system but written in the cache memory. In the disk read operation upon occurrence of the cache miss in read operation, since data transfer is started with a read enable sector, disk rotation need not be awaited and high-speed transfer can be performed.

An embodiment of a magnetic disk controller for using a cache memory regardless of the number of magnetic disk units will be described below. The same reference numerals in the embodiment of Fig. 17 denote the same parts as in the embodiment of Fig. 2, and a detailed description thereof will be omitted. In the embodiment of Fig. 17, magnetic disk units 10_1 through 10_4 are connected to HDC 21.

Fig. 18 shows a directory data format. The directory data is 4-byte data consisting of first to fourth bytes. The first byte consists of 3-bit drive number LUN (Logical Unit No.) as a magnetic disk drive identifier, and upper 5 bits (address U) of a 15-bit disk block address. The second byte consists of 8 bits (address M) of the disk block address which follow address U. The third byte consists of the remaining lower two bits (address L) of the disk block address, 4-bit all "0" data, one-bit error flag ERR, and one-bit valid display flag VB. The fourth byte consists of an 8-bit LRU (Least Recently Used) value. The directory data is produced on the basis of drive number LUN, the disk logical address, etc., included in the read/write command.

Fig. 19 shows a read/write command format. This command is a 6-byte instruction including a command code for designating the read/write operation, drive number LUN, the disk logical address, the number of data blocks, and cache control bits CC2 through CC0. Drive number LUN consists of three bits to access one of disk units 10_1 through 10_4. In this embodiment, the MSB of drive number LUN is always logic "0". The disk logical address represents a disk sector number designated by drive number LUN, and consists of a 5-bit logical address 2, 8-bit logical address 1, and 8-bit logical address 0, a total of 21 bits. In this embodiment, one block consists of 16 Kbytes. As shown in Fig. 19, bits 6 through 20 of the disk logical address represent a disk block address. In this embodiment, the capacity of cache memory 33 is 512 Kbytes, and one cache block consists of 16 Kbytes. The number of columns is 2, namely: column 0 and column 1. Each columns of cache memory 33 includes 16 cache blocks. Therefore, bit 6 of the disk logical address represents the column number.

Fig. 20 shows a structure of directory memory 35_1. The capacity of memory 35_1 is 128 bytes and consists of the column 0 area (64 bytes) starting with address 000 (hexadecimal), and the column 1 area (64 bytes) starting with address 040 (hexadecimal). The areas of columns 0 and 1 are subdivided into 16 areas (hereinafter, each called as to "level") each. Each level consists of 4 bytes. The directory data in Fig. 18 can be registered in each level.

The operation of the embodiment shown in Fig. 17 will be described below.

The read/write command (Fig. 19) from host system 11 is retained in command register 21_1 in HDC 21 through D/R 25 and bus 24. This command is read out from register 21_1 under the control of microprocessor 27, and is stored in a predetermined area of work memory 29. In this state, microprocessor 27 issues a seek command which is then set in HDC 21. HDC 21 controls a specified one of magnetic disk units 10_1 through

10_4 so as to seek the designated track.

Microprocessor 27 checks bit 6 (i.e., the column number) of the disk logical address, and checks if the disk block address including the disk logical address in the read/write command is registered in directory memory 35_1. If bit 6 is set at logic "0", the column 0 area of directory memory 35_1 serves as the area subjected to directory search. Microprocessor 27 sets "000" (hex) as the directory start address in CDMA 40.

However, if bit 6 is set at logic "1", the column 1 area of directory memory 35_1 serves as the area subjected to directory search. In this case, microprocessor 27 sets "040" (hex) in CDMA 40.

Microprocessor 27 sets drive number LUN and bits 16 through 20 (logical address 2) of the disk logical address in the read/write command in RTCA0 register 50 in register file 38. At the same time, microprocessor 27 sets bits 8 through 15 (logical address 1) of the disk logical address in RTCA1 register 51. Microprocessor 27 sets data consisting of bits 6 and 7 of the disk logical address, the 4-bit all "0" data, error flag ERR of logic "0", and valid display flag VB of logic "1" in RTCA2 register 52. Microprocessor 27 sets the LRU value (minimum LRU value) of 0 in RTCA3 register 53. Microprocessor 27 produces target data for directory search on the basis of drive number LUN and the disk logical address in the read/write command, and sets it in register file 38.

When the above-mentioned set-up is completed, microprocessor 27 causes cache control circuit 34 to perform directory search. The following directory search cycle is repeated 16 times.

In the directory search cycle, the directory memory address specified by CDMA 40 is set in RCGA0 register 60 in file 41. The directory data of a level represented by CDMA 40 is read out from memory 35_1 from the first byte to the fourth byte onto bus 31. The readout byte data is supplied to the B input of comparator 39. In synchronism with the read operation described above, the target data is read out from file 38 from the first byte in units of bytes. The readout target data is supplied to the A input of comparator 39. Comparator 39 compares the contents of the A and B inputs. The comparison result is supplied from comparator 39 to cache control circuit 34. The LRU value in the directory data supplied from directory memory 35 to the B input of comparator 39 through bus 31 is then loaded in CLRU 36 and incremented by one.

Cache control circuit 34 determines if cache hit occurs according to the comparison results for the first through third bytes in comparator 39. If cache hit is detected, control circuit 34 adds a cache hit flag to the directory memory address set in RCGA0 register 60 in file 41, i.e., in the address representing the start position of the corresponding level of the current directory search cycle. The address with the cache hit flag is then set in RCGA1 register 61.

Cache control circuit 34 determines the level of memory 35 which corresponds to the candidacy of the cache block to be replaced at this time, according to the LRU value comparison result of comparator 39. If the LRU value from directory memory 35 exceeds that from register file 38, control circuit 34 determines that the cache block corresponding to the level is a cache block to be replaced. In this case, control circuit 34 transfers the directory memory address from RCGA0 register 60 in file 41 to RCGA2 register 62. Control circuit 34 causes driver 37 to send the unitary incremented LRU value specified by CLRU 36 onto bus 31, and updates the LRU value set in register 53 in file 38 to the LRU value on bus 31.

Cache control circuit 34 writes the LRU value of CLRU 36 on bus 31 in the corresponding level of directory memory 35, regardless of the comparison result of comparator 39. In this embodiment, the LRU value in the corresponding level in directory memory 35 is incremented by one for every directory search cycle.

The above operation is the directory search cycle. The cycle is repeated for the respective directory data of levels 0 through F (hexadecimal) in directory memory 35, thereby completing directory search corresponding to the read/write command. In this case, cache hit is detected and the start address of the level corresponding to the cache block is set in RCGA1 register 61 in file 41. The start address of the level having the largest LRU value in the specified column is stored in RCGA2 register 62 in file 41. This address represents the directory memory 35 level corresponding to the cache block to be replaced if cache miss occurs in the read mode.

When the directory search is completed, microprocessor 27 fetches the content of RCGA1 register 61 in file 41 through bus 26, and checks the flag mentioned above if cache hit or miss occurs. If cache miss occurs in the read mode, microprocessor 27 writes the directory address of RCGA2 register 62 in file 41 as the oldest cache block at this time, i.e., the start address of the directory memory 35 level corresponding to the cache block to be replaced. This start address is set in CDMA 40. Microprocessor 27 registers the directory data in the level corresponding to directory memory 35 while the contents of CDMA 40 are sequentially counted. The directory data is substantially the same as the target data in register file 38, except that the LRU value is the initial value. In this embodiment, the initial value of the LRU value is specified by cache control bits CC2 through CC0.

As is apparent from the above description, since the directory data includes drive number LUN representing the magnetic disk drive, the block of the cache memory 33 to be replaced for a disk block in the disk units of this embodiment can be determined according to the magnitudes of the LRU values of the directory data for each level in the same manner as in the magnetic disk controller for controlling the single magnetic disk unit. Therefore, unlike the conventional system, the area of directory memory 35 can be assigned to any one of units 10_1 through 10_4, as shown in Fig. 21. Thus, in this embodiment, cache memory 33 can be effectively utilized according to the access frequency of units 10_1 through 10_4.

In the embodiment of Fig. 17, cache memory 33 is commonly used by magnetic disk units 10_1 through 10_4. However, for a magnetic disk unit having a disk block such as an operating system which is loaded once, the cache memory need not be used. In this case, the cache memory is commonly used by other, remaining drive units. Thus, cache memory 33, having a limited capacity, can be effectively used, and the hit rate can be increased. Whether the cache memory is used is determined by a command at the time of system initialization in units of magnetic disk drive units.

The above embodiments exemplify the disk cache system for a magnetic disk. However, the present invention can also be applied to disk cache systems such as floppy disk drive units, optical disk systems or RAM disk units.

Claims

1. A magnetic disk controller incorporating a cache system having a cache memory for storing a plurality of cache blocks and adopting an LRU (Least Recently Used) scheme as a replacement algorithm for the cache blocks, characterized in that there are provided:

determining means (35) for determining if data requested on the basis of the disk access request from a host system is stored in said cache memory (33);

cache operation designation data storage means (45) for storing cache operation designation data consisting of a cache memory address, the number of transfer sectors, a data transfer direction, and data representing necessity or lack of necessity of data transfer to said host system;

sector number reading means (27) for reading a sector number stored in an ID field of a sector earliest read, at a given moment, when a read miss is determined by said determining means upon a read request from said host system;

cache operation designation data producing means (27) for producing the cache operation designation data on the basis of contents of the disk access request from said host system and a determination result from said determining means, and for storing the cache operation designation data in said cache operation designation data storage means (45), said cache operation designation data producing means (27) being adapted, in case of a read miss; to produce the cache operation designation data for each one of subunits obtained such that the cache block to be replaced is divided by a cache memory address corresponding to a sector next to that read by said sector number reading means (27), a cache address corresponding to a start sector of a disk area, data of which is to be transferred to said host system, and a cache memory address corresponding to a last sector thereof; and to sequentially store the cache operation designation data respectively corresponding to the subunits of said cache block in said cache operation designation data storage means, the cache operation designation data including, as a transfer start cache address, the cache address corresponding to the sector next to that the number of which is read by said reading means (27) being adapted to serve as start cache operation designation data; and

data transfer means (27, 21, 34) for reading out the cache operation designation data from said cache operation designation data storage means (45) in an order starting from the start cache operation designation data, and for performing data transfer operation on the basis of the readout data.

2. A magnetic disk controller incorporating a cache system having a cache memory for storing a plurality of cache blocks and adopting an LRU (Least Recently Used) scheme as a replacement algorithm for the cache blocks, wherein said cache system comprises:

a directory memory (35_1) having fields storing, in respective entries corresponding to respective cache blocks of said cache memory, designation information for blocks registered in said cache memory and LRU counter fields indicating resident status in said cache memory of these blocks;

command issuing means (28) for issuing a read/write command to which an arbitrary LRU setting value and designation information for a read/write target block are appended;

directory search means (35_2) for searching whether the read/write target block is regis-

tered in said cache memory by comparing the registered block designation information of respective entries with the read/write designation information; and

registering means (27) for registering the arbitrary LRU setting value (93) appended to the read/write command in a hit entry of said directory memory when hit occurs, and in the LRU counter field of an entry corresponding to the cache block in which the read/write target block is registered when a miss hit occurs,

characterized in that said magnetic disk controller further comprises:

determining means (35) for determining if data requested on the basis of the disk access request from a host system is stored in said cache memory (33);

cache operation designation data storage means (45) for storing cache operation designation data consisting of a cache memory address, the number of transfer sectors, a data transfer direction, and data representing necessity or lack of necessity of data transfer to said host system;

sector number reading means (27) for reading a sector number stored in an ID field of a sector earliest read, at a given moment, when a read miss is determined by said determining means upon a read request from said host system;

cache operation designation data producing means (27) for producing the cache operation designation data on the basis of contents of the disk access request from said host system and a determination result from said determining means, and for storing the cache operation designation data in said cache operation designation data storage means (45), said cache operation designation data producing means (27) being adapted, in case of a read miss; to produce the cache operation designation data for each one of subunits obtained such that the cache block to be replaced is divided by a cache memory address corresponding to a sector next to that read by said sector number reading means (27), a cache address corresponding to a start sector of a disk area, data of which is to be transferred to said host system, and a cache memory address corresponding to a last sector thereof; and to sequentially store the cache operation designation data respectively corresponding to the subunits of said cache block in said cache operation designation data storage means, the cache operation designation data including, as a transfer start cache address, the cache address corresponding to the sector next to that the number of which is read by said reading means (27) being adapted to serve as start cache operation designation data; and

data transfer means (27, 21, 34) for reading out the cache operation designation data from said cache operation designation data storage means (45) in an order starting from the start cache operation designation data, and for performing data transfer operation on the basis of the readout data.

3. A controller according to claim 1 or 2, characterized in that the data representing the data transfer direction and the necessity or lack of necessity of data transfer to said host system presents five modes, consisting of:

i) a mode used for read hit, in which the contents of said cache memory (33) are transferred to said host system and said magnetic disk unit is not accessed;

ii) a mode for read miss, in which said magnetic disk unit is read-accessed, the read data is transferred to said host system and, at the same time, is written in said cache memory;

iii) a mode for read miss, in which data of a disk area not accessed by said host system must be read so that the read data from said magnetic disk unit is written in said cache memory but is not transferred to said host system;

iv) a mode for write hit, in which the write data from said host system is transferred to said magnetic disk unit and, at the same time, is written in said cache memory; and

v) a mode for write miss, in which the data from said host system is not written in said cache memory but in said magnetic disk unit.

FIG. 1

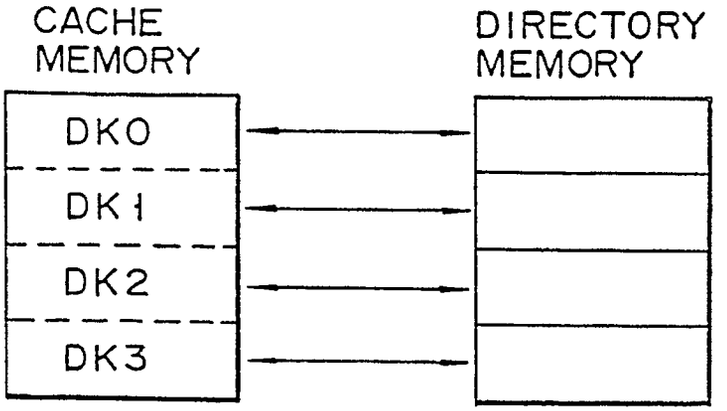


FIG. 3

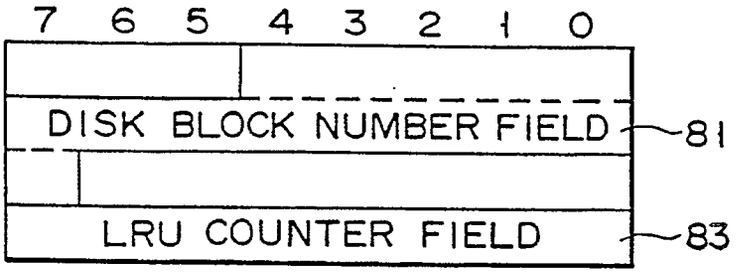
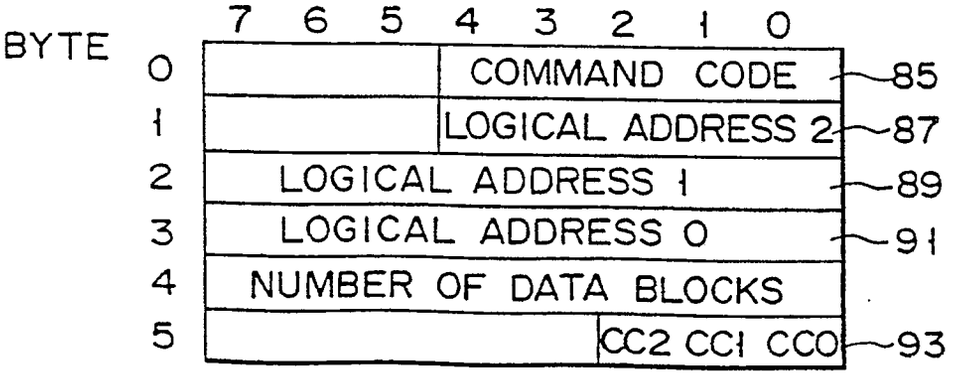


FIG. 4



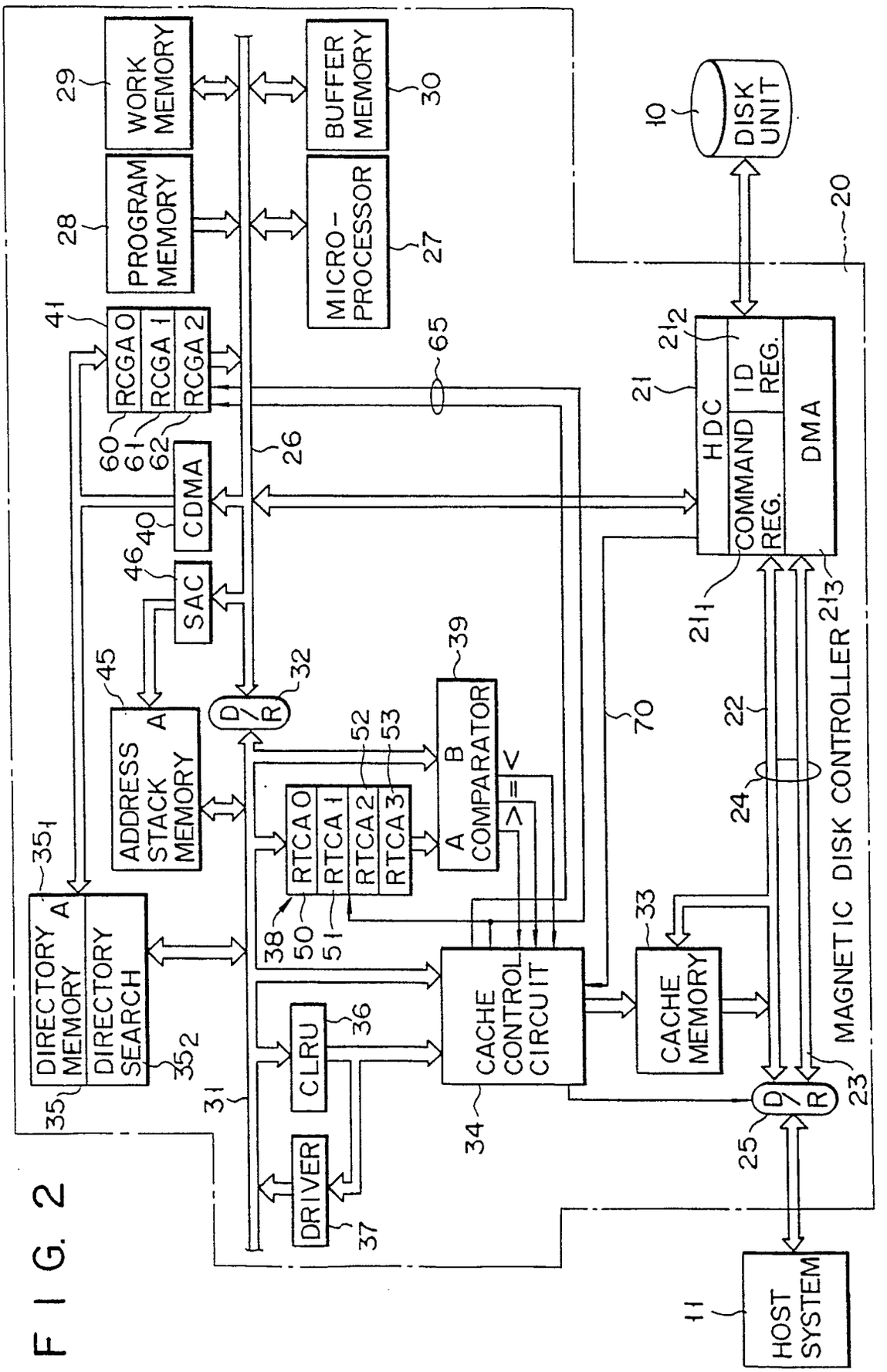

WORK MEMORY
29
BUFFER MEMORY
30
PROGRAM MEMORY
28
MICRO-PROCESSOR
27
DISK UNIT
10
20
41
RCGA 0
RCGA 1
RCGA 2
60
61
62
65
26
CDMA
40
SAC
46
HDC
COMMAND REG.
ID REG.
DMA
21
212
211
213
45
ADDRESS STACK MEMORY
A
D
R
32
39
B
A
COMPARATOR
>
=
<
52
53
RTCA 0
RTCA 1
RTCA 2
RTCA 3
38
50
51
70
22
24
351
DIRECTORY MEMORY
DIRECTORY SEARCH
352
35
31
CLRU
36
DRIVER
37
CACHE CONTROL CIRCUIT
34
33
CACHE MEMORY
MAGNETIC DISK CONTROLLER
23
25
HOST SYSTEM
11

FIG. 2

FIG. 5

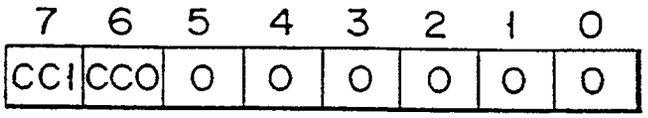


FIG. 6

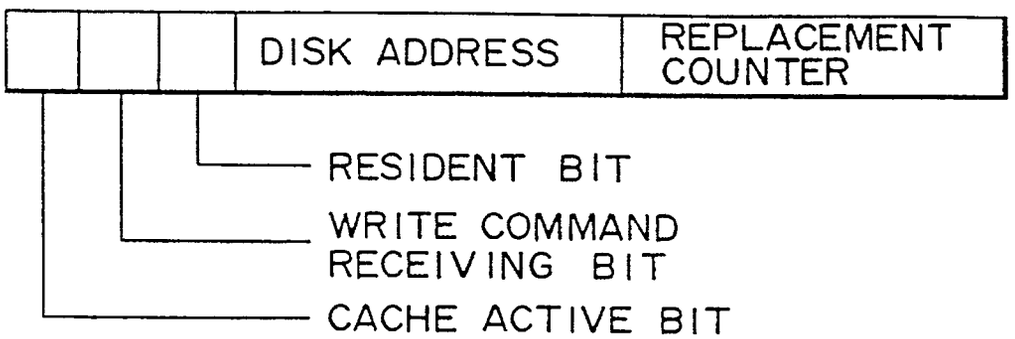


FIG. 7

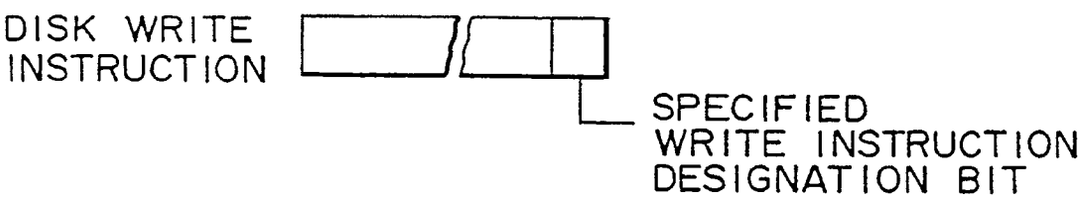


FIG. 8

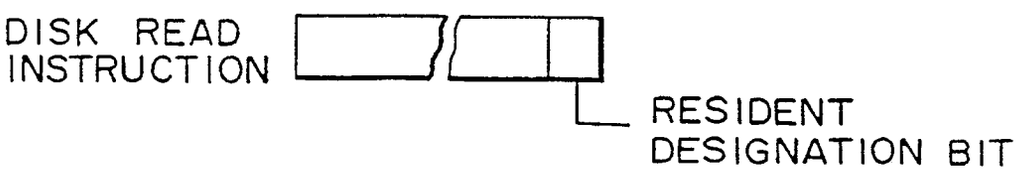


FIG. 9

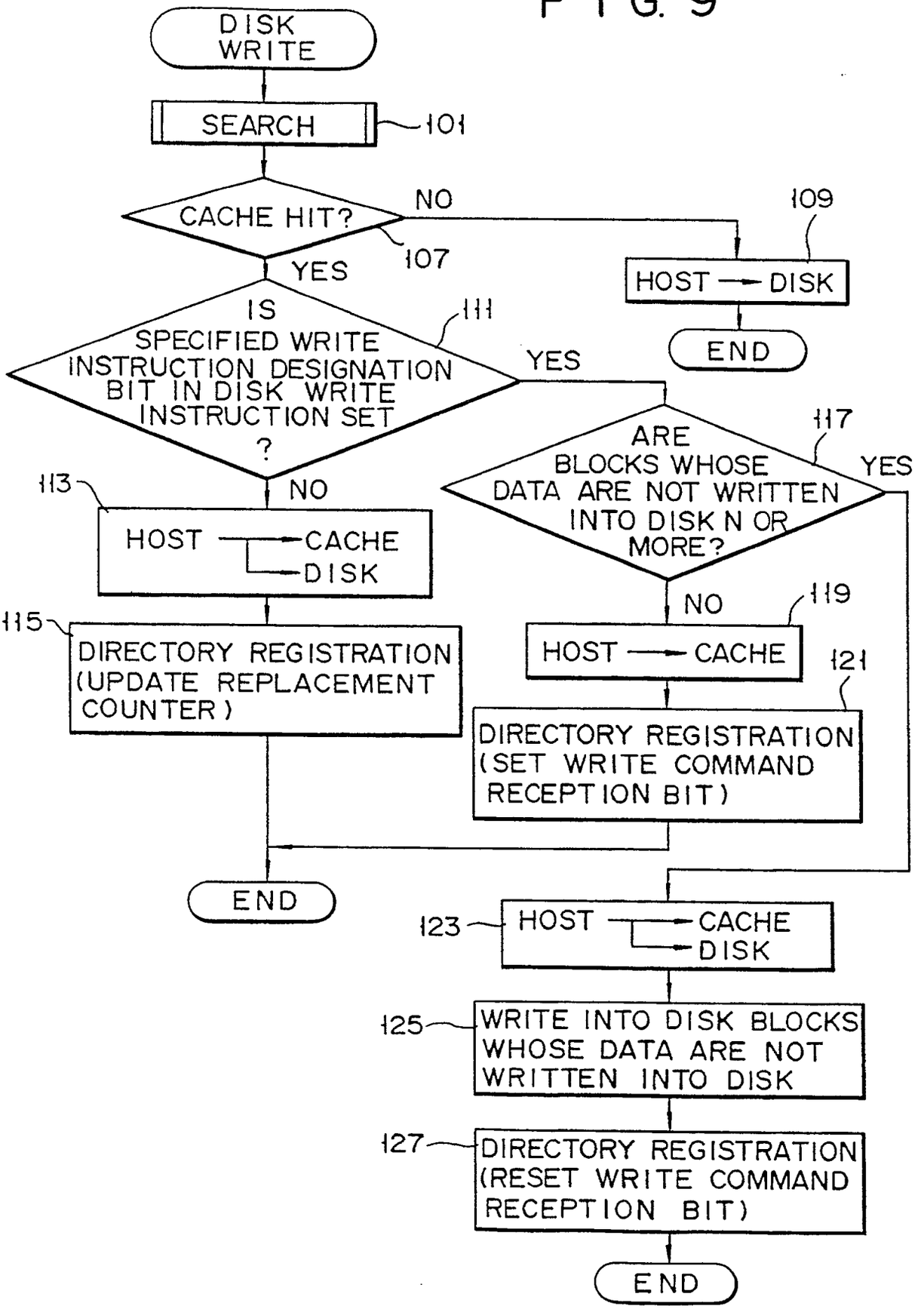

DISK WRITE
SEARCH
101
CACHE HIT?
NO
107
109
HOST → DISK
END
YES
IS SPECIFIED WRITE INSTRUCTION DESIGNATION BIT IN DISK WRITE INSTRUCTION SET ?
111
YES
ARE BLOCKS WHOSE DATA ARE NOT WRITTEN INTO DISK N OR MORE?
117
YES
113
NO
HOST → CACHE, DISK
115
DIRECTORY REGISTRATION (UPDATE REPLACEMENT COUNTER)
NO
119
HOST → CACHE
121
DIRECTORY REGISTRATION (SET WRITE COMMAND RECEPTION BIT)
END
123
HOST → CACHE, DISK
125
WRITE INTO DISK BLOCKS WHOSE DATA ARE NOT WRITTEN INTO DISK
127
DIRECTORY REGISTRATION (RESET WRITE COMMAND RECEPTION BIT)
END

FIG. 10

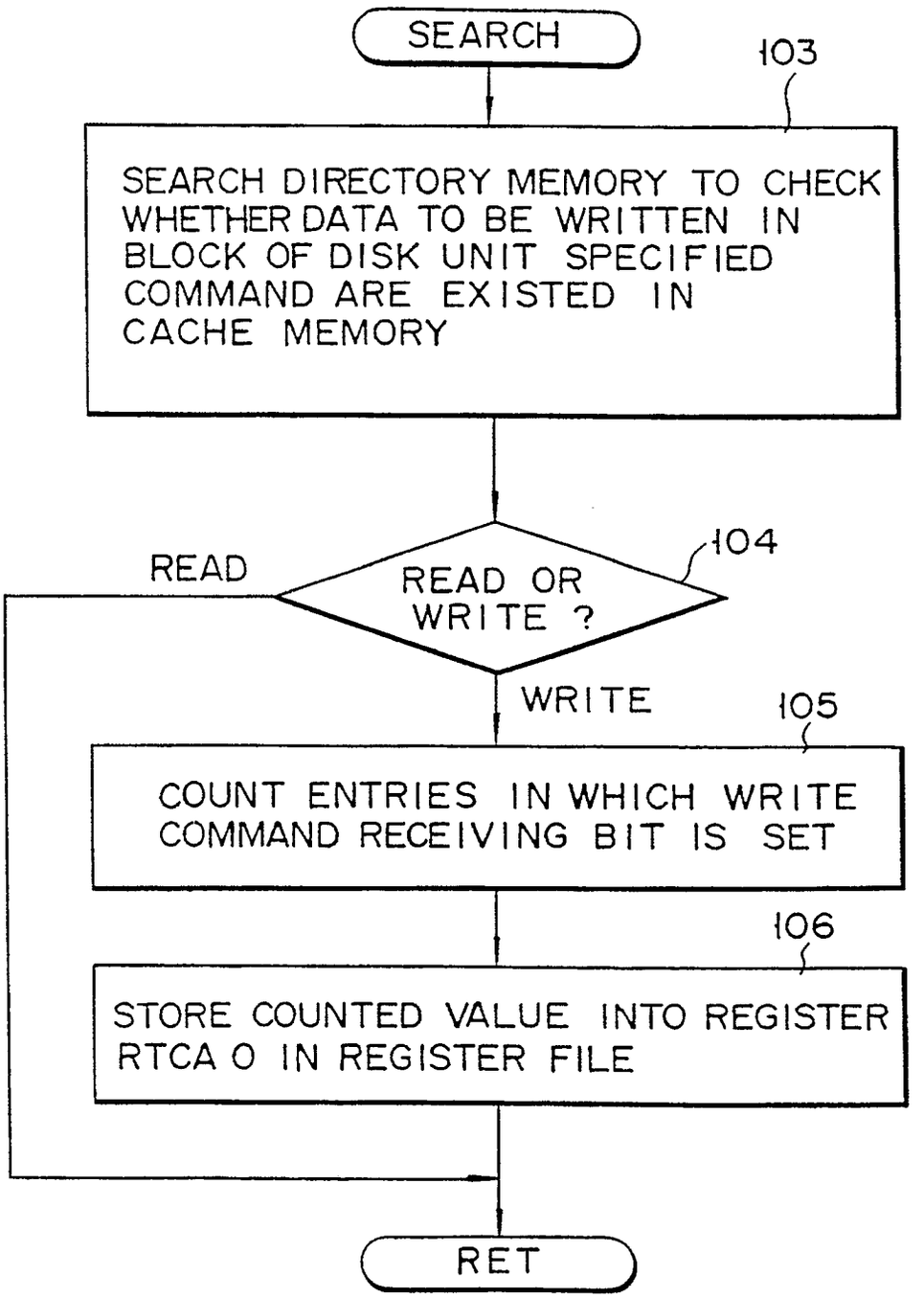

SEARCH
103
SEARCH DIRECTORY MEMORY TO CHECK WHETHER DATA TO BE WRITTEN IN BLOCK OF DISK UNIT SPECIFIED COMMAND ARE EXISTED IN CACHE MEMORY
104
READ OR WRITE ?
READ
WRITE
105
COUNT ENTRIES IN WHICH WRITE COMMAND RECEIVING BIT IS SET
106
STORE COUNTED VALUE INTO REGISTER RTCA 0 IN REGISTER FILE
RET

FIG. 11

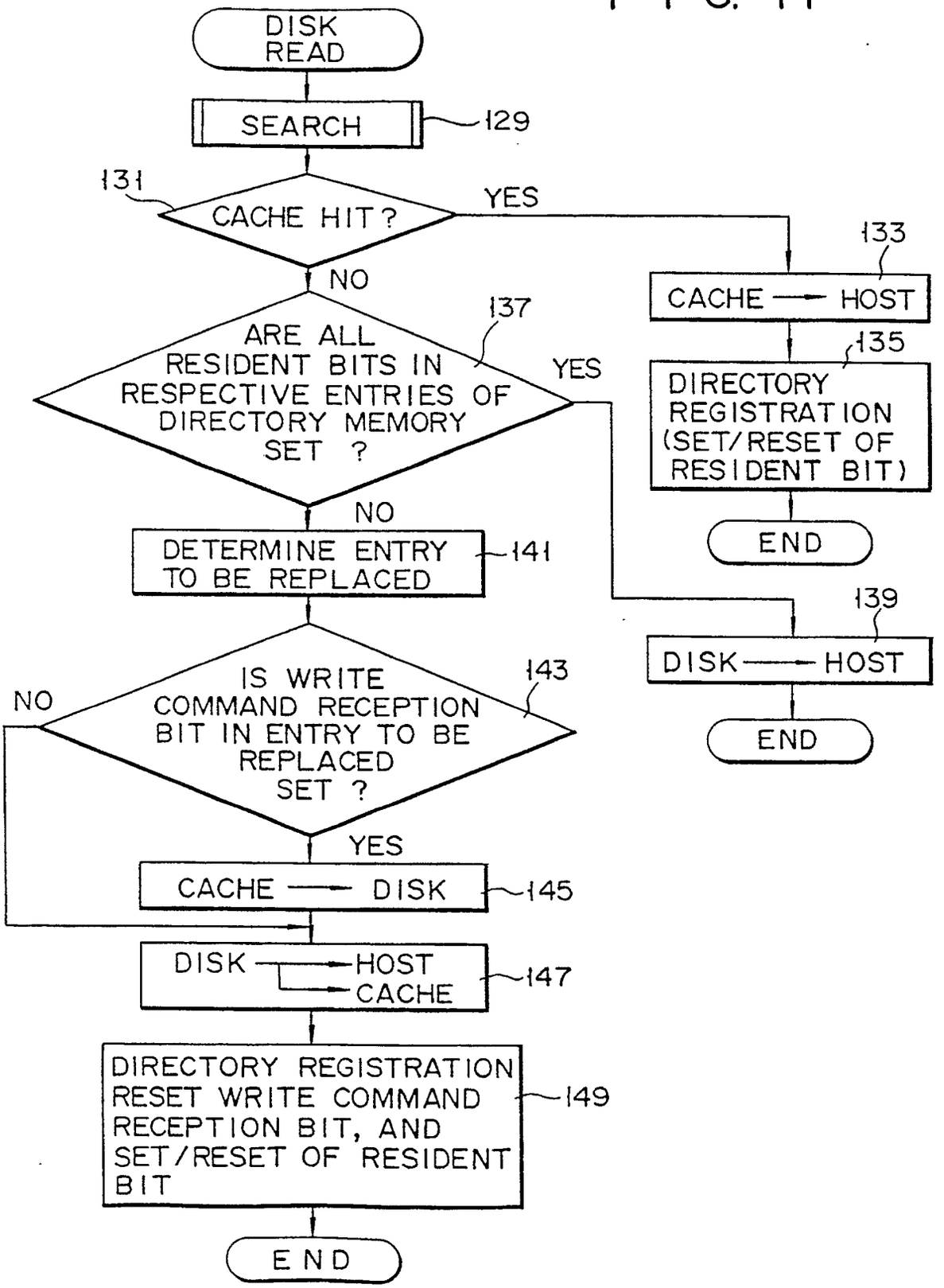

DISK READ
SEARCH
129
131
CACHE HIT?
YES
133
CACHE → HOST
NO
137
ARE ALL RESIDENT BITS IN RESPECTIVE ENTRIES OF DIRECTORY MEMORY SET ?
YES
135
DIRECTORY REGISTRATION (SET/RESET OF RESIDENT BIT)
END
NO
DETERMINE ENTRY TO BE REPLACED
141
139
DISK → HOST
END
143
NO
IS WRITE COMMAND RECEPTION BIT IN ENTRY TO BE REPLACED SET ?
YES
CACHE → DISK
145
DISK → HOST
CACHE
147
DIRECTORY REGISTRATION RESET WRITE COMMAND RECEPTION BIT, AND SET/RESET OF RESIDENT BIT
149
END

FIG. 12

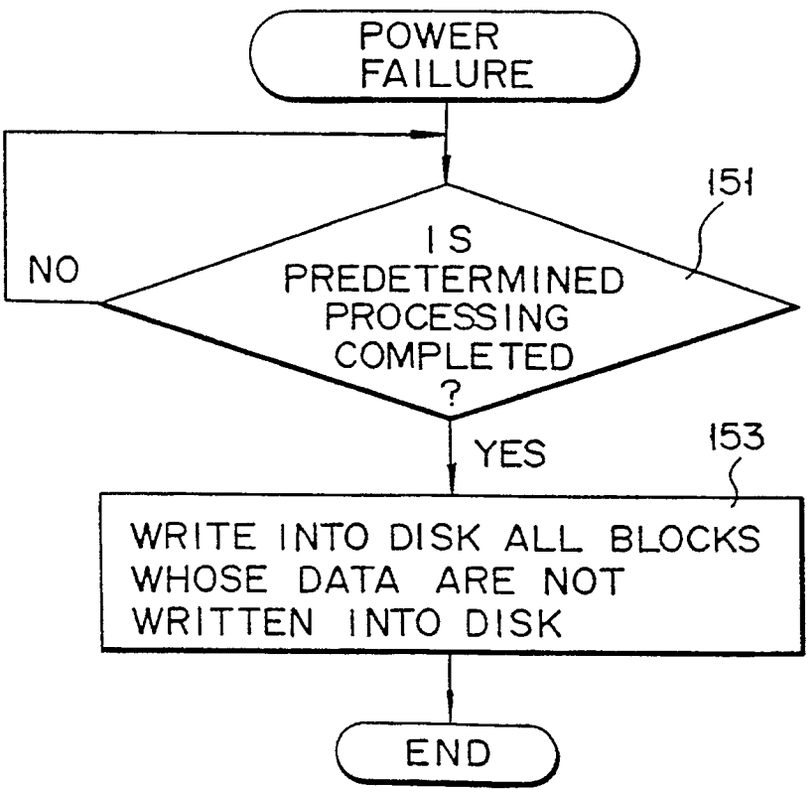

POWER FAILURE
151
IS PREDETERMINED PROCESSING COMPLETED ?
NO
YES
153
WRITE INTO DISK ALL BLOCKS WHOSE DATA ARE NOT WRITTEN INTO DISK
END

FIG. 13A

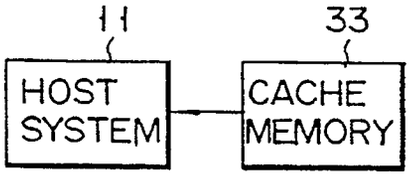


FIG. 13B

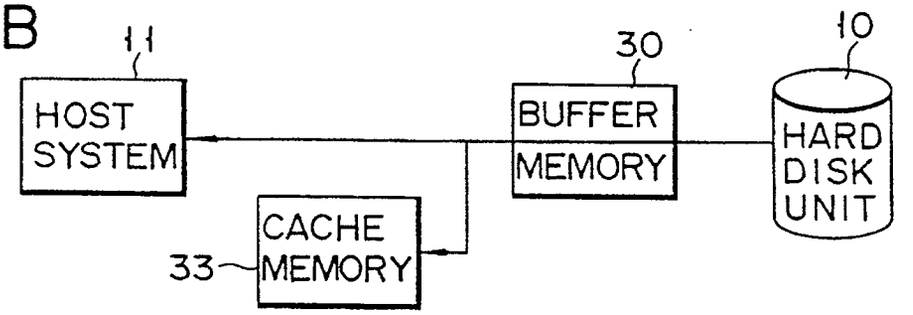


FIG. 13C

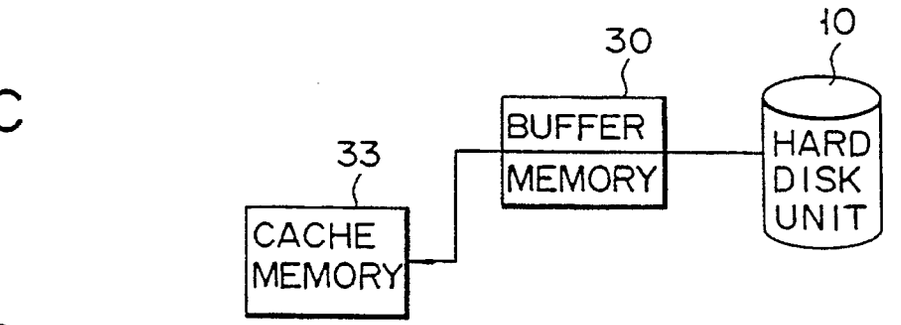


FIG. 13D

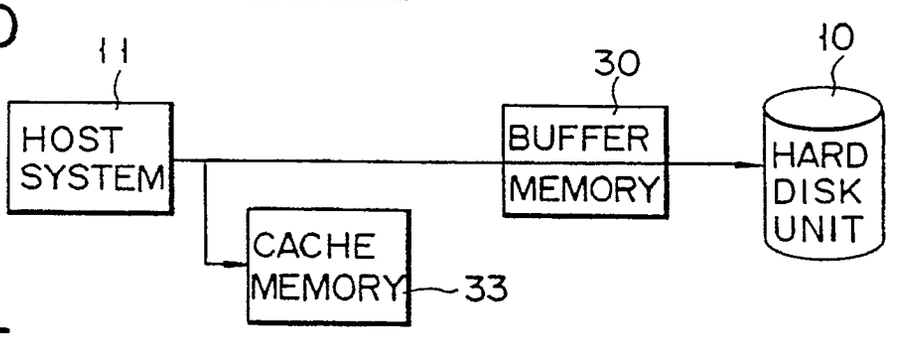


FIG. 13E

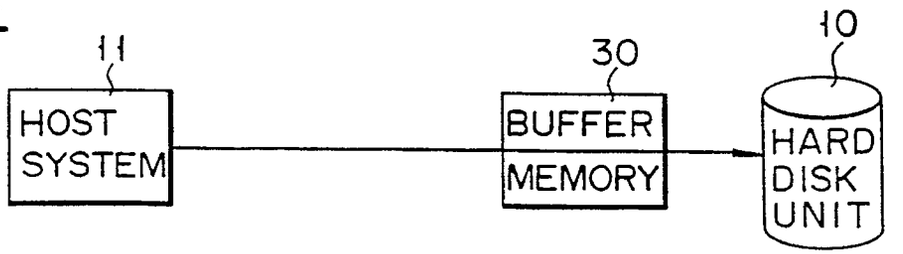


FIG. 14

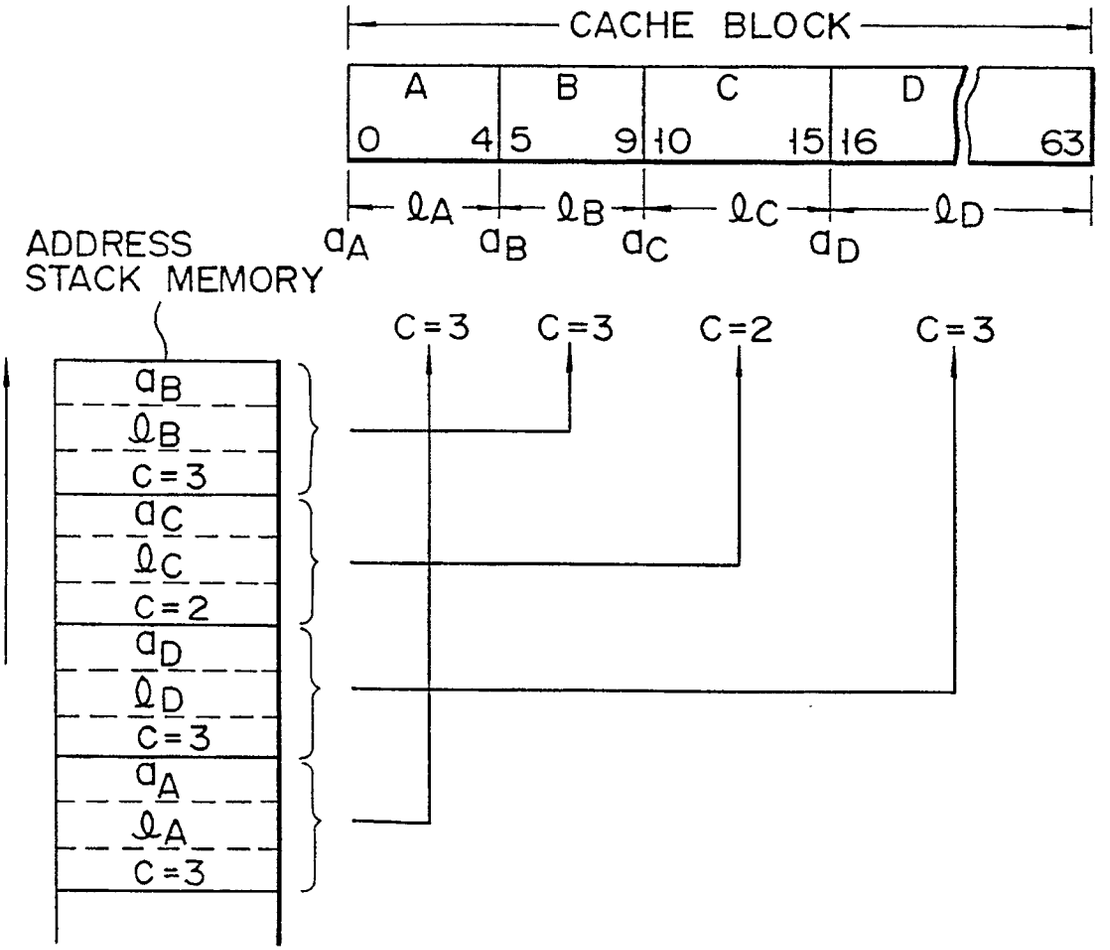

CACHE BLOCK
A
B
C
D
0
4
5
9
10
15
16
63
ℓ_A
ℓ_B
ℓ_C
ℓ_D
a_A
a_B
a_C
a_D
ADDRESS
STACK MEMORY
C=3
C=3
C=2
C=3
a_B
ℓ_B
C=3
a_C
ℓ_C
C=2
a_D
ℓ_D
C=3
a_A
ℓ_A
C=3

FIG. 15

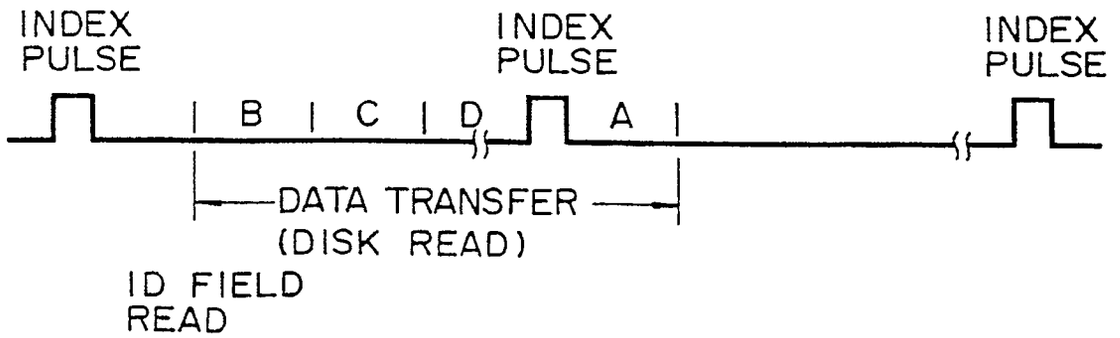
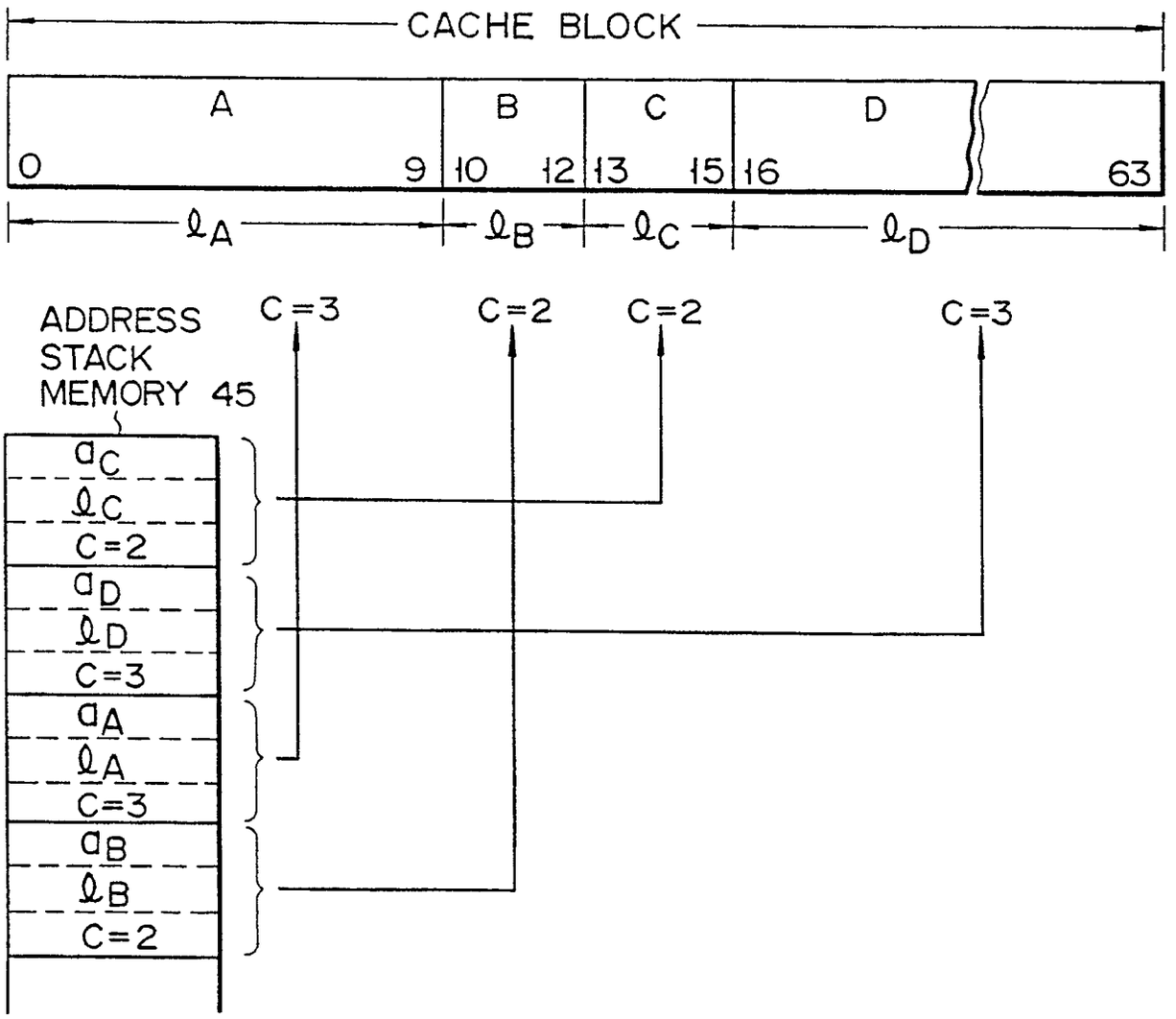

INDEX
PULSE
INDEX
PULSE
INDEX
PULSE
B
C
D
A
DATA TRANSFER
(DISK READ)
ID FIELD
READ

FIG. 16


CACHE BLOCK
A
B
C
D
0
9
10
12
13
15
16
63
ℓ_A
ℓ_B
ℓ_C
ℓ_D
ADDRESS STACK MEMORY 45
C=3
C=2
C=2
C=3
a_C
ℓ_C
C=2
a_D
ℓ_D
C=3
a_A
ℓ_A
C=3
a_B
ℓ_B
C=2

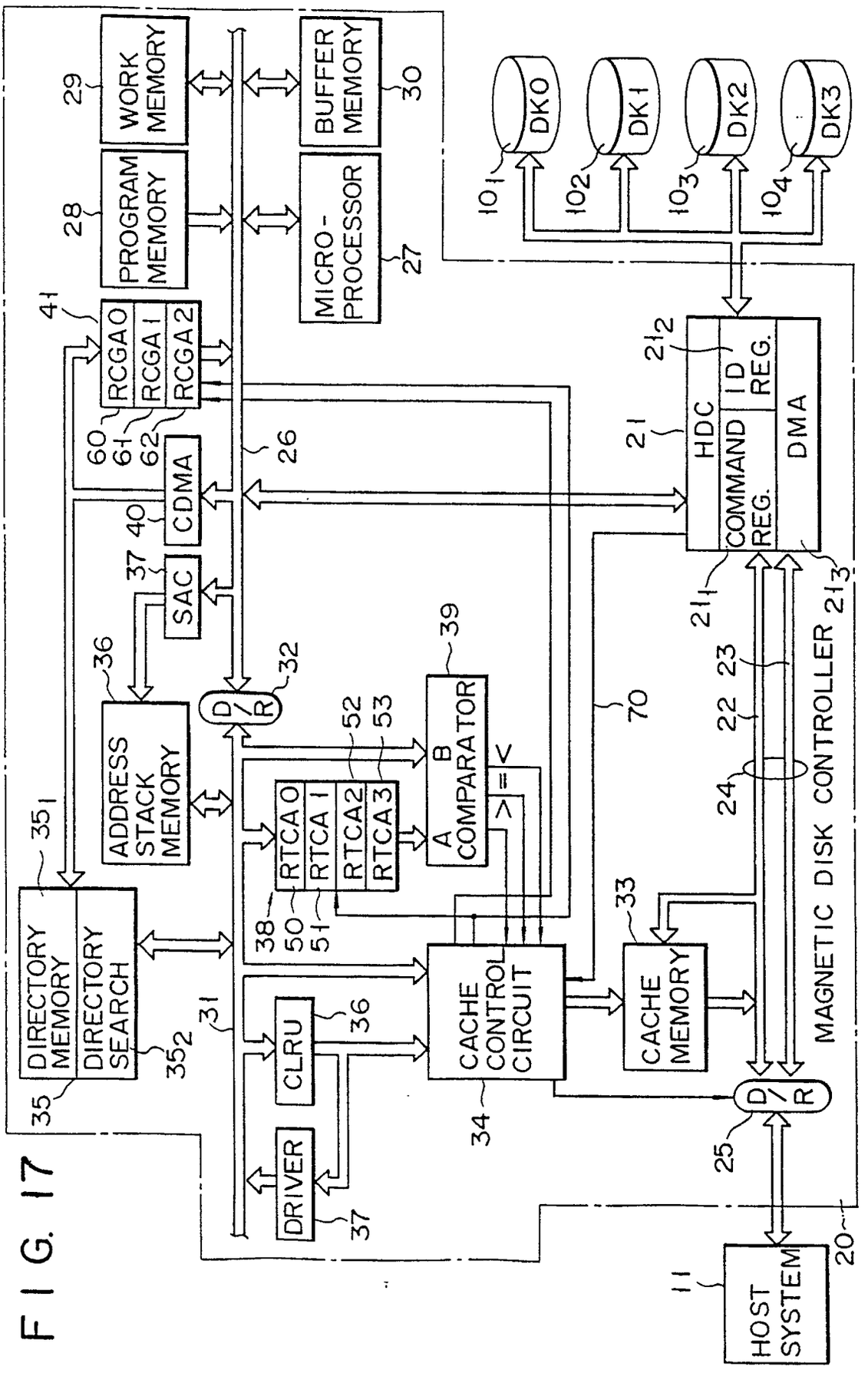

WORK MEMORY
29
PROGRAM MEMORY
28
BUFFER MEMORY
30
MICRO-PROCESSOR
27
DK0
DK1
DK2
DK3
10 1
10 2
10 3
10 4
41
RCGA0
RCGA1
RCGA2
60
61
62
CDMA
40
26
SAC
37
36
ADDRESS STACK MEMORY
D/R
32
39
B COMPARATOR
A
V
=
>
52
53
RTCA0
RTCA1
RTCA2
RTCA3
38
50
51
35 1
DIRECTORY MEMORY
DIRECTORY SEARCH
35 2
35
31
CLRU
36
DRIVER
37
CACHE CONTROL CIRCUIT
34
33
CACHE MEMORY
70
HDC
COMMAND REG.
ID REG.
DMA
21
21 2
21 1
21 3
22
23
24
MAGNETIC DISK CONTROLLER
D/R
25
20
HOST SYSTEM
11

FIG. 17

FIG. 18

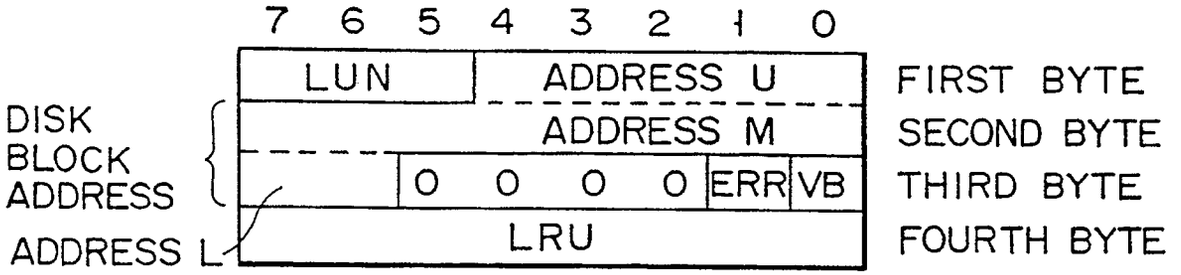

7 6 5 4 3 2 1 0
LUN
ADDRESS U
ADDRESS M
0 0 0 0 ERR VB
LRU
DISK BLOCK ADDRESS
ADDRESS L
FIRST BYTE
SECOND BYTE
THIRD BYTE
FOURTH BYTE

FIG. 19

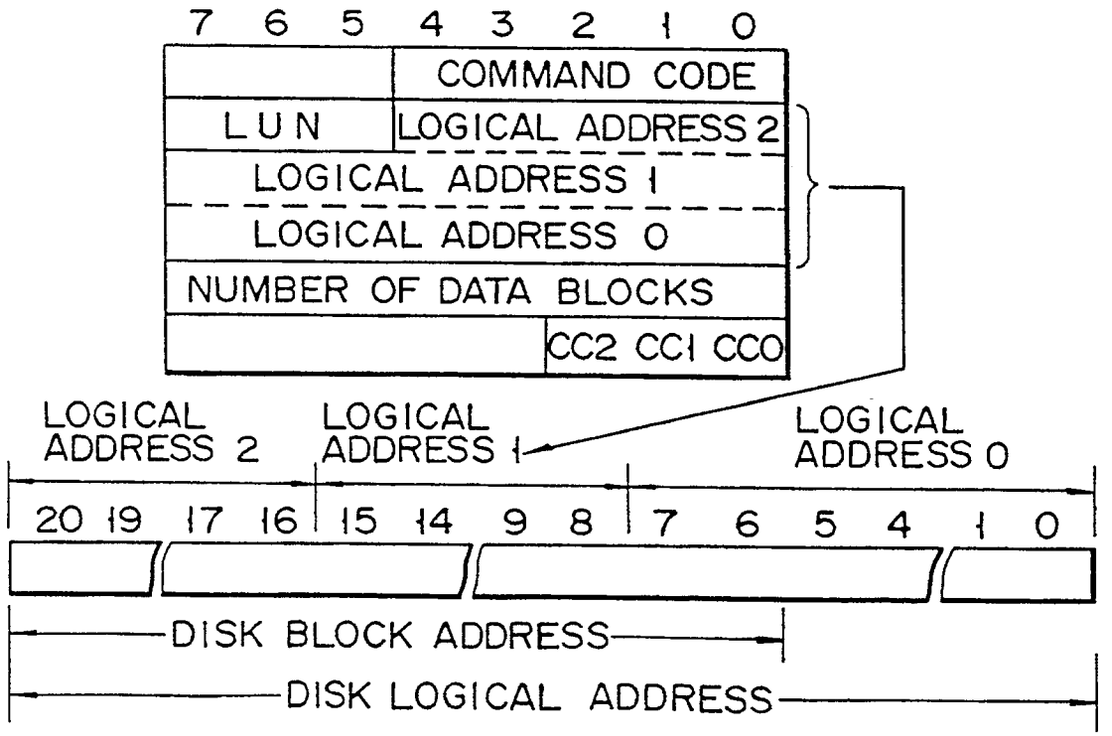

7 6 5 4 3 2 1 0
COMMAND CODE
LUN
LOGICAL ADDRESS 2
LOGICAL ADDRESS 1
LOGICAL ADDRESS 0
NUMBER OF DATA BLOCKS
CC2 CC1 CC0
LOGICAL ADDRESS 2
LOGICAL ADDRESS 1
LOGICAL ADDRESS 0
20 19 17 16 15 14 9 8 7 6 5 4 1 0
DISK BLOCK ADDRESS
DISK LOGICAL ADDRESS

FIG. 20

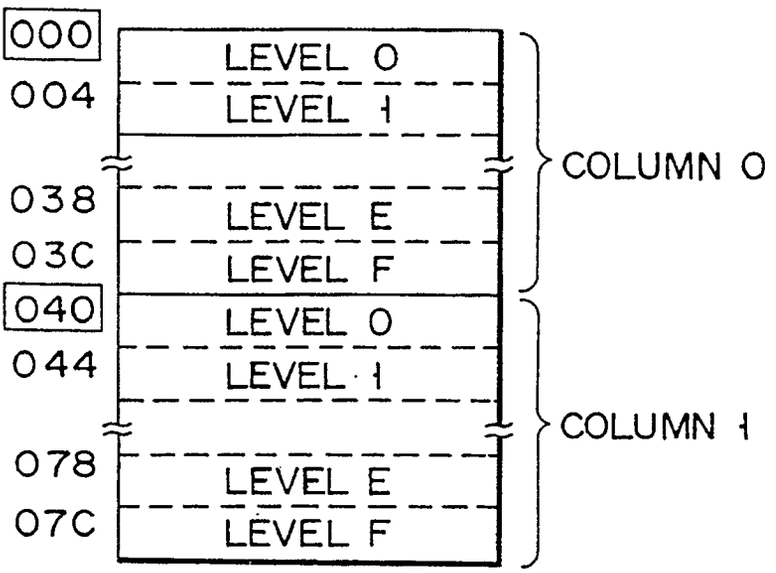
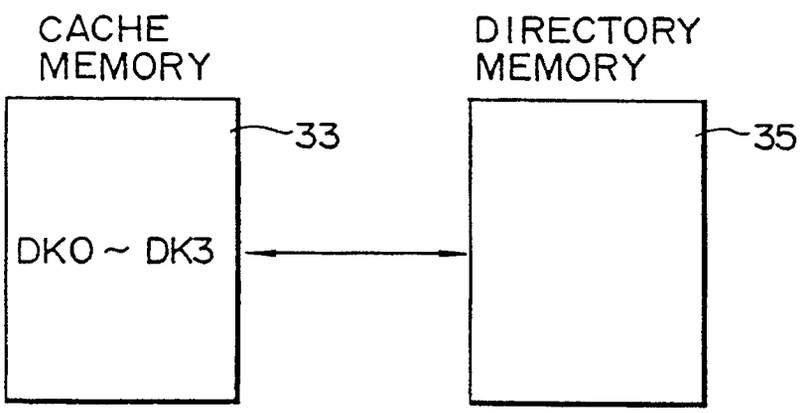

000
004
038
03C
040
044
078
07C
LEVEL 0
LEVEL 1
LEVEL E
LEVEL F
LEVEL 0
LEVEL 1
LEVEL E
LEVEL F
COLUMN 0
COLUMN 1

FIG. 21


CACHE MEMORY
DIRECTORY MEMORY
33
35
DK0 ~ DK3